

## APPENDIX

### ADDINDEX()

#### **Description**

Add index(es) to Customer DNA system.

#### **Input Parameters**

eiaCd

CustomerDataSeq (see cdna.idl for structure)

#### **Return Values**

void

#### **Exceptions Thrown**

ErrorB::CDNAException

Note: If at least one system key exists in the cross-reference table, the other index(es) may be added to the cross-reference table. The customer data may not be processed because there is no guarantee that it is more accurate or complete than the existing customer data.

#### **Procedure**

- 1.0 Validate EIA
- 2.0 For CustomerSeq.length
  - 2.1 If customer.key already exists
    - 2.1.1 Write to error log, continue with next customer
  - 2.2 If customer does not have minimum fields
    - 2.2.1 Write to error log, continue with next customer
  - 2.3 convertOut = TB::convert(CustomerSeq[n] )
  - 2.4 parseIn = (selected fields from) convertOut
  - 2.5 parseOut = TB::parse(parseIn)
  - 2.6 geoCountry = parseOut.country
  - 2.7 geocodeOut = TB::geocode(parseOut, geoCountry)
  - 2.8 matchData = (selected fields from ) geocodeOut
  - 2.9 createWindow as follows:
    - 2.9.1 if matchData.fqtv
      - 2.9.1.1 fqtvS = Pfqtv::queryFqtv(matchData.fqtv)
      - 2.9.1.2 custs = Pcust::query(fqtv[n].cdnald)
      - 2.9.1.3 for each custs
        - 2.9.1.3.1 windowItem[next].fields = cust[n].fields // assuming cust  
!null
        - 2.9.1.3.2 addToWindow(windowItem)
    - 2.9.2 if matchData.email
      - 2.9.2.1 emails = Pemail::queryEmailAddr(matchData.email)
      - 2.9.2.2 custs = Pcust::query(email[n].cdnald)

```

2.9.2.2.1 windowItem[next].fields = cust[n].fields // assuming cust
2.9.2.2.2 addToWindow(windowItem)
2.9.3 Repeat as above with document, creditcard, phone
2.9.4 If matchData.candCd does not contain spaces (address available)
2.9.4.1 cands = Pcand::queryCandCd(matchData.candCd)
2.9.4.2 custs = Pcust::query(cand[n].cdnald)
2.9.4.2.1 windowItem[next].fields = cust[n].fields
2.10 matchOut = match(matchData, window)
2.11 if no match
2.11.1 pcustomer = (selected fields from) matchOut
2.11.2 cdnald = Pcustomer::add(pcustomer)
2.12 if more than one match
2.12.1 matched = select best one (based on criteria tbd)
2.12.2 cdnald = matched.cdnald
2.13 else (only one matched)
2.13.1 cdnald = matched.cdnald
2.14 todayDate = Time::today()
2.15 PXRef::add(cdnald, clientsystemkey, todayDate)
3.0 Return

```

## **DELETEINDEX()**

### **Description**

Delete customer index(es) to Customer DNA system. If the last reference to a specific customer, then this method may invoke the deleteCustomer method.

### **Input Parameters**

eiaCd

SystemKeySeq // See cdna.idl for structure

### **Return Values**

void

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

- 1.0 Validate Eia
- 2.0 For systmKeySeq.length
  - 2.1 xref = Pxref::queryKey(systemKey[n])
  - 2.2 cdnald = xref.cdnald
  - 2.3 Pxref::delete(xref)
  - 2.4 Xrefs = Pxref::queryKey(cdnald)
  - 2.5 If no xrefs returned
    - 2.5.1 Pcust.delete(cdnald)
- 3.0 Return successful

FOOTNOTES

#### **LAW OFFICES**

INNEGAN, HENDERSON,  
FARABOW, GARRETT,  
& DUNNER, L.L.P.  
1300 I STREET, N. W.  
WASHINGTON, DC 20005  
202-408-4000

## **GETINDEX()**

### **Description**

Retrieve the Indexes to a specific customer

### **Input Parameters**

eiaCd

SystemKeyData

### **Return Values**

SystemKeyDataSeq

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

- 1.0 Validate Eia
- 2.0 xref = Pxref.queryKey(SystemKey)
- 3.0 xrefs = Pxref.queryKey(xref.cdnaId)
- 4.0 systemKeyData[n] = xrefs[n].systemKeyData
- 5.0 return systemKeyData

2007-04-23 12:00:00

## **VALIDEIA**

### **Description**

Validates the external interfacing application (EIA) code

### **Input Parameters**

EiaCd

### **Return Values**

bool

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

- 1.0 Initialized Indicator = FALSE
- 2.0 eia\_var = eiaFactory->query(eiaCd)
- 3.0 If (!PS\_is\_nil())
  - 3.1 Indicator = TRUE
- 4.0 Return Indicator

09004-0700  
"48F20660"

#### **LAW OFFICES**

FINNEGAN, HENDERSON,  
FARABOW, GARRETT,  
& DUNNER, L.L.P.  
1300 I STREET, N. W.  
WASHINGTON, DC 20005  
202-408-4000

## **VALIDDOCTYPE()**

### **Description**

Validates the documentation type before allowing a document to be added.

### **Input Parameters**

docType

### **Return Values**

bool

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

- 1.0 Initialized Indicator = FALSE
- 2.0 docType\_var = docTypeFactory->query(docType)
- 3.0 If (!PS\_is\_nil())
  - 3.1 Indicator = TRUE
- 4.0 Return Indicator

2007-04-24 09:00

## **MODIFYCUSTOMER()**

### **Description**

Update customer data.

### **Input Parameters**

eiaCd

SystemKeyData

Customer

### **Return Values**

void

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

- 1.0 Validate Eia
- 2.0 cdnald = Pxref::queryKey(systemKeyData)
- 3.0 If no record, throw error exception
- 4.0 oldCust = Pcust::query(custId)
- 5.0 If address or phone fields exists in customer (fields to update include address or phone fields)
  - 5.1 ConvertIn = (fields from) customer
  - 5.2 convertOut = TB::convert(convertIn )
  - 5.3 parseIn = (selected fields from) convertOut
  - 5.4 parseOut = TB::parse(parseIn)
  - 5.5 geoCountry = parseOut.country
  - 5.6 geocodeOut = TB::geocode(parseOut, geoCountry)
  - 5.7 cdnald =Pxref::queryKey( systemKeyData ) // Find existing customer record
  - 5.8 oldCust.address = geocodeOut.address
  - 5.9 oldCust.phone = geocodeOut.phone (// do we add the phone or replace??)
- 6.0 Pemail::add(customer.emailAddr) // Do only if customer.emailAddr is not null , of course..
- 7.0 Pother::add(customer.other) // for creditcard, document etc.
- 8.0 createWindow as follows:
  - 8.1 if matchData.fqtv
    - 8.1.1 fqtvS = Pfqtv::queryFqtv(matchData.fqtv)
    - 8.1.2 custs = Pcust::query(fqtv[n].cdnald)
    - 8.1.3 for each custs
      - 8.1.3.1 windowItem[next].fields = cust[n].fields // assuming cust !=null
      - 8.1.3.2 addToWindow(windowItem)
  - 8.2 if matchData.email
    - 8.2.1 emails = Pemail::queryEmailAddr(matchData.email)
    - 8.2.2 custs = Pcust::query(email[n].cdnald)

LAW OFFICES

NEGAN, HENDERSON,  
ARABOW, GARRETT,  
& DUNNER, L.L.P.  
300 I STREET, N. W.  
WASHINGTON, DC 20005  
202-408-4000

8.2.2.1 windowItem[next].fields = cust[n].fields // assuming cust  
 8.2.2.2 addToWindow(windowItem)  
 8.3 Repeat as above with document, creditcard, phone  
 8.4 If matchData.candCd does not contain spaces (address available)  
   8.4.1 cand = Pcand::queryCandCd(matchData.candCd)  
   8.4.2 custs = Pcust::query(cand[n].cdnald)  
     8.4.2.1 windowItem[next].fields = cust[n].fields  
 9.0 matchOut = match(matchData, window)  
 10.0 if no match  
   10.1 pcustomer = (selected fields from) matchOut  
   10.2 cdnald = Pcust::add(pcustomer)  
 11.0 if more than one match  
   11.1 matched = select best one (based on criteria tbd)  
   11.2 cdnald = matched.cdnald  
 12.0 else (only one matched)  
   12.1 cdnald = matched.cdnald  
 13.0 todayDate = Time::today()  
 14.0 PXRef::add(cdnald, clientsystemkey, todayDate)  
 15.0 Return successful

100720-4872060



## **TCONVERT()**

### **Description**

This is a general purpose driver to encapsulate formatting and manipulating the data and files for the Trillium Converter callable module

### **Input Parameters**

customerData

### **Return Values**

convertOut

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

This module will prepare the various buffers for the Converter. The convertOut is the output from the converter callable module.

2007.02.14 13:20:50

## **TPARSE()**

### **Description**

This is a general purpose driver to encapsulate formatting and manipulating the data and files for the Trillium Parser callable module

### **Input Parameters**

parseIn

### **Return Values**

parseOut

### **Exceptions Thrown**

ErrorB::CDNAException

### **Procedure**

This module will prepare the various buffers for the Parser. The parseOut is the output from the converter callable module.

700720"48420653

Parameter	Value	Unit
Temperature	25.0	°C
Pressure	1.0	atm
Flow rate	1.0	L/min
Concentration	0.1	g/L
pH	7.0	
Time	1.0	h
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	L
Mass	0.1	g
Area	1.0	cm <sup>2</sup>
Length	1.0	cm
Width	1.0	cm
Height	1.0	cm
Depth	1.0	cm
Radius	1.0	cm
Diameter	1.0	cm
Volume	1.0	

This module will prepare the various buffers for the geocoder. The geocodeOut is the output from the geocoder callable module. There are requirements specific per country.

## CUSTOMER DNA IDL

```
//=====
// File:  cdna.idl (Customer DNA IDL)
// Description: Interface specification for the CDNA services
//=====
```

```
module CDNA
{
    struct Address
    {
        string stAddr1;
        string stAddr2;
        string geoline1; // For city, state, zip, country etc
        string geoline2;

        // Null flags
        boolean stAddr1NULL;
        boolean stAddr2NULL;
        boolean geoline1NULL;
        boolean geoline2NULL;
    };

    struct Document
    {
        string cryCode;
        string type;
        string number;
    };

    struct SystemKeyData
    {
        string client;
        string clientSystemKey;
    };

    typedef sequence <SystemKeyData> SystemKeyDataSeq;

    struct CDNACustomer
    {
        SystemKeyData key;
        string      name;
        Address      address;
        string      fqtvNr;
        string      phone;
        string      creditCard;
        string      emailAddr;
        Document     document;
    };
}
```

```

// Null flags
// NOTE: no nameNULL - name is mandatory;
// At least one of the following must be false (i.e. present)
boolean addressNULL;
boolean fqtvNrNULL;
boolean phoneNULL;
boolean creditCardNULL;
boolean emailAddrNULL;
boolean documentNull;

```

```

};
typedef sequence <CDNACustomer> CDNACustomerSeq;

```

```

struct TCustomer // To be standardized by Trillium only
{
    string    name;
    Address   address;
    // Null flags
    // NOTE: no null flags - name and address is mandatory
};

```

```

typedef sequence <TCustomer> TCustomerSeq;

```

```

typedef sequence <string> standardizedCustSeq;

```

```

// -----
// Interface Begin
// -----

```

```

exception CDNAException
{
    unsigned short code;
    string dateTime;
    string name;
    string desc;
};

```

```

interface CDNASession;
interface CDNASessionFactory;

```

```

interface CDNASessionFactory
{
    CDNASession create(in string name);
};

```

```

interface CDNASession
{

```

```

// Add a cross-reference item
//(with optional additional system keys)
void addCDNAXref(in string eiaCd, in CDNACustomerSeq customers)
raises (CDNAException);

// Deleting cross reference item
void deleteCDNAXref (in string eiaCd, in SystemKeyDataSeq keys)
raises (CDNAException);

// Customer data maintenance
void modifyCustomerData(in string eiaCd,
                        in CDNACustomerSeq customers)
raises (CDNAException);

// Retrieve cross-reference items for a customer
SystemKeyDataSeq getCDNAXref(in string eiaCd,
                             in SystemKeyData key)
raises (CDNAException);

// Convert, Parse, and Geocode customer name and address
standardizedCustSeq standardize(in string eiaCd,
                                in TCustomerSeq tCustomers)
raises (CDNAException);

```

```

};
};

```

2001-01-04 14:20:00